Hit List

Clear | Generate Collection | Print | Fwd Refs | Bkwd Refs | Generate OACS

Search Results - Record(s) 1 through 39 of 39 returned.

☐ 1. Document ID: US 20040109004 A1

Using default format because multiple data bases are involved.

L20: Entry 1 of 39 File: PGPB Jun 10, 2004

PGPUB-DOCUMENT-NUMBER: 20040109004
PGPUB-FILING-TYPE: new
DOCUMENT-IDENTIFIER: US 20040109004 A1

TITLE: Depth-of-field effects using texture lookup

PUBLICATION-DATE: June 10, 2004

INVENTOR-INFORMATION:

NAME	CITY	STATE	COUNTRY	RULE-47
Bastos, Rui M.	Santa Clara	CA	US	
Lew, Stephen D.	Sunnyvale	CA	US	
Beeson, Curtis A.	Fremont	CA	US	
Demers, Joseph E. JR.	Palo Alto	CA	US	

US-CL-CURRENT: 345/587

Full | Title | Citation | Front | Review | Classification | Date | Reference | Sequences | Attachments | Claims | KWIC | Draw. D

☐ 2. Document ID: US 20040085321 A1

L20: Entry 2 of 39 File: PGPB May 6, 2004

DOCUMENT-IDENTIFIER: US 20040085321 A1
TITLE: Game system with graphics processor

Detail Description Paragraph:
[0114] Texture mapping 323 is performed on the pixels in the pipeline, if this option has been activated. Texture mapping is in essence the "painting" of a bitmap texture onto a polygon. Texture mapping 323 for graphics processor 5 is shown in greater detail in FIG. 13. The color of a given pixel written to the frame buffer is determined by a combination of a texel color and the pixel color derived from the rasterization process. The texel color is determined from either the S,T,Q or U,V. These coordinates both refer to a texture map, a bitmapped image which contains texels (texture pixels) that are to be painted onto the polygon.

h e b b g e e e f e f ef b e

Full | Title | Citation | Front | Review | Classification | Date | Reference | Sequences | Attachments | Claims | KWIC | Draw De

☐ 3. Document ID: US 20040051716 A1

L20: Entry 3 of 39 File: PGPB Mar 18, 2004

DOCUMENT-IDENTIFIER: US 20040051716 A1
TITLE: Image processing

Summary of Invention Paragraph:
[0007] A functional overview of the components A7 to A10 of graphics accelerator A6 is shown in Figure B, in which a processing function of graphics processor A7 is to sort graphic data sent from processor A2 between three-dimensional data and two-dimensional data. Typically, 3-D data is sent by processor A7 as control points equipped with co-ordinates in a volume configured with a Cartesian co-ordinate system to a first evaluating system sub-processor A81, where vertices are evaluated from said control points. Said vertices are then sent from said sub-processor A81 to a second vertex processing sub-processor A82, where they are converted into primitives, i.e. polygons. Two-dimensional data is sent by processor A7 to a third texture-processing sub-processor A83, a typical function of which is to generate positional data and color attributes, such that when the polygons are sent from sub-processor A82 to a fourth rasterizing sub-processor A84 for conversion into screen-positioned, two-dimensional pixels, said two-dimensional graphic data is similarly rasterized at a correct position within the eventual image frame and, when said pixels are sent from said sub-processor A84 to a fifth fragment-processing sub-processor A85 for further color and/or transparency data processing, the two-dimensional graphic data-dependent color attributes are correctly processed and associated with said pixels. In accordance with the description of Figure A, sub-processor A85 outputs final image frame data to frame buffer A10 but may optionally loop said output back to said texture-processing sub-processor A83 in order to reprocess said output, for instance if data processing operations performed by said sub-processor A85 require multiple passes.

Full | Title | Citation | Front | Review | Classification | Date | Reference | Sequences | Attachments | Claims | KWIC | Draw De

☐ 4. Document ID: US 20040042654 A1

L20: Entry 4 of 39 File: PGPB Mar 4, 2004

DOCUMENT-IDENTIFIER: US 20040042654 A1
TITLE: Accelerated matte

Summary of Invention Paragraph:
[0008] A functional overview of the components A7 to A10 of graphics accelerator A6 is shown in FIG. B, in which a processing function of graphics processor A7 is to sort graphic data sent from processor A2 between three-dimensional data and two-dimensional data. Typically, 3-D data is sent by processor A7 as control points equipped with co-ordinates in a volume configured with a Cartesian co-ordinate system to a first evaluating system sub-processor A81, where vertices are evaluated

h e b b g e e e f e f ef b e

from said control points. Said vertices are then sent from said sub-processor A81 to a second vertex processing sub-processor A82, where they are converted into primitives, i.e. polygons. Two-dimensional data is sent by processor A7 to a third texture-processing sub-processor A83, a typical function of which is to generate positional data and color attributes, such that when the polygons are sent from sub-processor A82 to a fourth rasterizing sub-processor A84 for conversion into screen-positioned, two-dimensional pixels, said two-dimensional graphic data is similarly rasterized at a correct position within the eventual image frame and, when said pixels are sent from said sub-processor A84 to a fifth fragment-processing sub-processor A85 for further color and/or transparency data processing, the two-dimensional graphic data-dependent color attributes are correctly processed and associated with said pixels. In accordance with the description of FIG. A, sub-processor A85 outputs final image frame data to frame buffer A10 but may optionally loop said output back to said texture-processing sub-processor A83 in order to reprocess said output, for instance if data processing operations performed by said sub-processor A85 require multiple passes.

☐ 5. Document ID: US 20040041820 A1

L20: Entry 5 of 39 File: PGPB Mar 4, 2004

DOCUMENT-IDENTIFIER: US 20040041820 A1
TITLE: Image processing

Summary of Invention Paragraph:
[0007] A functional overview of the components A7 to A10 of graphics accelerator A6 is shown in Figure B, in which a processing function of graphics processor A7 is to sort graphic data sent from processor A2 between three-dimensional data and two-dimensional data. Typically, 3-D data is sent by processor A7 as control points equipped with co-ordinates in a volume configured with a Cartesian co-ordinate system to a first evaluating system sub-processor A81, where vertices are evaluated from said control points. Said vertices are then sent from said sub-processor A81 to a second vertex processing sub-processor A82, where they are converted into primitives, i.e. polygons. Two-dimensional data is sent by processor A7 to a third texture-processing sub-processor A83, a typical function of which is to generate positional data and color attributes, such that when the polygons are sent from sub-processor A82 to a fourth rasterizing sub-processor A84 for conversion into screen-positioned, two-dimensional pixels, said two-dimensional graphic data is similarly rasterized at a correct position within the eventual image frame and, when said pixels are sent from said sub-processor A84 to a fifth fragment-processing sub-processor A85 for further color and/or transparency data processing, the two-dimensional graphic data-dependent color attributes are correctly processed and associated with said pixels. In accordance with the description of Figure A, sub-processor A85 outputs final image frame data to frame buffer A10 but may optionally loop said output back to said texture-processing sub-processor A83 in order to reprocess said output, for instance if data processing operations performed by said sub-processor A85 require multiple passes.

Full | Title | Citation | Front | Review | Classification | Date | Reference | Sequences | Attachments | Claims | KWIC | Draw De

h e b b g e e e f e f ef b e

6. Document ID: US 20030142103 A1

L20: Entry 6 of 39 File: PGPB Jul 31, 2003

DOCUMENT-IDENTIFIER: US 20030142103 A1
TITLE: Method and apparatus for rasterizing in a hierarchical tile order

Summary of Invention Paragraph:
[0005] During the rasterization stage, the graphics processor renders each primitive into the frame buffer. The graphics processor accomplishes this task by determining which frame buffer memory locations are included within the bounds of each primitive. The included memory locations are then initialized to reflect the attributes of the primitive, including color and texture.

Summary of Invention Paragraph:
[0017] The graphics processor rasterizes graphics primitives into the frame buffer. To accomplish this task, the graphics processor determines which frame buffer memory locations are included within the bounds of each primitive. The included memory locations are then initialized to reflect the attributes of the primitive, including color and texture. During rasterization, the graphics processor uses a hierarchy of memory tiles. Within this hierarchy, smaller tiles are grouped into larger tiles. These larger tiles may be grouped, in turn, into still larger tiles. For a representative embodiment of the present invention, the tile hierarchy includes three levels. The lowest level of the hierarchy is made up of four pixel by four pixel low-level tiles. These four-by-four tiles are grouped into eight-by-eight mid-level tiles and the eight-by-eight tiles are grouped into sixteen-by-sixteen high-level tiles.

7. Document ID: US 20030067473 A1

L20: Entry 7 of 39 File: PGPB Apr 10, 2003

DOCUMENT-IDENTIFIER: US 20030067473 A1
TITLE: Method and apparatus for executing a predefined instruction set

Summary of Invention Paragraph:
[0004] To process video graphics data, particularly three dimensional (3D) graphics, the central processing unit executes video graphics or geometric software to produce geometric primitives, which are often triangles. A plurality of triangles is used to generate an object for display. Each triangle is defined by a set of vertices, where each vertex is described by a set of attributes. The attributes for each vertex can include spatial coordinates, texture coordinates, color data, specular color data or other data as known in the art. Upon receiving a geometric primitive, the raster engine of the video graphics processor generates pixel data based on the attributes for one or more of the vertices of the primitive. The generation of pixel data may include, for example, texture mapping operations performed based on stored textures and texture coordinate data for each of the vertices of the primitive. The pixel data generated is blended with the current contents of the frame buffer such that the contribution of the primitive being rendered is included in the display frame. Once the raster engine has

generated pixel data for an entire frame, or field, the pixel data is retrieved from the frame buffer and provided to the display.

☐ 8. Document ID: US 20030030643 A1

L20: Entry 8 of 39 File: PGPB Feb 13, 2003

DOCUMENT-IDENTIFIER: US 20030030643 A1
TITLE: Method and apparatus for updating state data

Summary of Invention Paragraph:
[0003] To process video graphics data, particularly 3D graphics, the central processing unit executes video graphics or geometric software to produce geometric primitives, which are often triangles. A plurality of triangles is used to generate an object for display. Each triangle is defined by a set of vertices, where each vertex is described by a set of attributes. The attributes for each vertex can include spatial coordinates, texture coordinates, color data, specular color data or other data as known in the art. Upon receiving a geometric primitive, a transform and lighting engine (or vertex shader engine) of the video graphics processor may convert the data from 3D to projected two-dimensional (2D) coordinates and apply coloring and texture coordinate computations to the vertex data. Thereafter, the raster engine of the video graphics processor generates pixel data based on the attributes for one or more of the vertices of the primitive. The generation of pixel data may include, for example, texture mapping operations performed based on stored textures and texture coordinate data for each of the vertices of the primitive. The pixel data generated is blended with the current contents of the frame buffer such that the contribution of the primitive being rendered is included in the display frame. Once the raster engine has generated pixel data for an entire frame, or field, the pixel data is retrieved from the frame buffer and provided to the display.

Full | Title | Citation | Front | Review | Classification | Date | Reference | Sequences | Attachments | Claims | KWIC | Draw De

---

☐ 9. Document ID: US 20030016229 A1

L20: Entry 9 of 39 File: PGPB Jan 23, 2003

DOCUMENT-IDENTIFIER: US 20030016229 A1
TITLE: Methods and apparatus for memory management

Summary of Invention Paragraph:
[0004] During the rasterization stage, the graphics processor renders each primitive into the frame buffer. The graphics processor accomplishes this task by determining which frame buffer memory locations are included within the bounds of each primitive. The included memory locations are then initialized to reflect the attributes of the primitive, including color and texture.

h e b b g e e e f e f ef b e

Full | Title | Citation | Front | Review | Classification | Date | Reference | Sequences | Attachments | Claims | KWIC | Draw De

☐ 10. Document ID: US 20020033827 A1

L20: Entry 10 of 39 File: PGPB Mar 21, 2002

DOCUMENT-IDENTIFIER: US 20020033827 A1
TITLE: Graphic processor and data processing system

Detail Description Paragraph:
[0060] The rendering buffer unit 101 is used as a buffer between the unified memory unit 4 external to the graphic processor 1 and the rendering unit 100. To be more specific, data read out from the memory-buffer area of the unified memory unit 4 and data to be written into the memory-buffer area is temporarily stored in the rendering buffer unit 101. In addition, information, such as a command and a texture, read out from the unified memory unit 4 are also temporarily stored in the rendering buffer unit 101. On the other hand, the display buffer unit 103 is used as a buffer between the unified memory unit 4 and the display unit 102. To be more specific, image information read out from the frame-buffer area of the unified memory unit 4 is temporarily stored in the display buffer unit 103 before being passed on to the display unit 102. The display buffer unit 103 has a color pallet 1031 for converting image data of 1 pixel per 8 bits into image data of 1 pixel per 16 bits.

Full | Title | Citation | Front | Review | Classification | Date | Reference | Sequences | Attachments | Claims | KWIC | Draw De

☐ 11. Document ID: US 20010039556 A1

L20: Entry 11 of 39 File: PGPB Nov 8, 2001

DOCUMENT-IDENTIFIER: US 20010039556 A1
TITLE: Digital filter

Detail Description Paragraph:
[0070] Next, the blending (mixed mode processing) function of the graphic processor 110 will be described. The graphic processor 110 effects the blending function to calculate the blending between an output color (source color) Cs after the texture mapping and a pixel color (destination color) Cd in the frame buffer region.

Full | Title | Citation | Front | Review | Classification | Date | Reference | Sequences | Attachments | Claims | KWIC | Draw De

☐ 12. Document ID: US 6747661 B1

L20: Entry 12 of 39 File: USPT Jun 8, 2004

h e b b g e e e f e f ef b e

DOCUMENT-IDENTIFIER: US 6747661 B1
TITLE: Graphics data compression method and system

Detailed Description Text (6):
A memory controller 80 coupled to the pixel engine 78 and the graphics processor 70 handles memory requests to and from the host memory 22, and a local memory 84. The local memory 84 stores graphics data, such as texture data, in the compressed format provided by the data compression circuit 76 and the graphics processor 70, and additionally stores both source pixel color values and destination pixel color values. Destination color values are stored in a frame buffer (not shown) within the local memory 84. In a preferred embodiment, the local memory 84 is implemented using random access memory (RAM), such as dynamic random access memory (DRAM), or static random access memory (SRAM). A display controller 88 coupled to the local memory 84 and to a first-in first-out (FIFO) buffer 90 controls the transfer of destination color values stored in the frame buffer to the FIFO 90. Destination values stored in the FIFO 90 are provided to a digital-to-analog converter (DAC) 92, which outputs red, green, and blue analog color signals to the display 46 (FIG. 3).

13. Document ID: US 6734867 B1

L20: Entry 13 of 39 File: USPT May 11, 2004

DOCUMENT-IDENTIFIER: US 6734867 B1
TITLE: Cache invalidation method and apparatus for a graphics processing system

Detailed Description Text (5):
A pixel engine 78 is coupled to receive the graphics data generated by the graphics processor 70, as well as an ID number assigned by the graphics processor 70 to blocks of graphics data stored in the host memory 18. Use of the ID numbers by the graphics processing system 40 will be explained in greater detail below. The pixel engine 78 contains circuitry for performing various graphics functions, such as, but not limited to, texture application, bilinear filtering, fog, blending, color space conversion, and dithering. A memory controller 80 coupled to the pixel engine 78 and the graphics processor 70 handles memory requests to and from the host memory 18, and a local memory 84. The local memory 84 stores both source pixel color values and destination pixel color values. Destination color values are stored in a frame buffer (not shown) within the local memory 84. In a preferred embodiment, the local memory 84 is implemented using random access memory (RAM), such as dynamic random access memory (DRAM), or static random access memory (SRAM). A display controller 88 coupled to the local memory 84 and to a first-in first-out (FIFO) buffer 90 controls the transfer of destination color values stored in the frame buffer to the FIFO 90. Destination values stored in the FIFO 90 are provided to a digital-to-analog converter (DAC) 92, which outputs red, green, and blue analog color signals to the display 46 (FIG. 1).

h e b b g e e e f e f ef b e

☐ 14. Document ID: US 6677967 B2

L20: Entry 14 of 39 File: USPT Jan 13, 2004

DOCUMENT-IDENTIFIER: US 6677967 B2
TITLE: Video game system for capturing images and applying the captured images to animated game play characters

CLAIMS:

1. In a software-controlled home video game machine system specifically designed for interactive 3D video game play, including 3D animated graphics and associated sound generation, said home video game machine system including a user-operable hand-held controller having a housing and a joystick provided thereon communicating with (i) a main processor, (ii) a 3D graphics coprocessor connected to the main processor for providing at least polygon coordinate transformation and light source processing, and (iii) at least one memory including a frame buffer communicating with the 3D graphics coprocessor, said at least one memory storing plural polygon coordinates defining surfaces of 3D animated video game characters, the home video game machine system playing interactive games based on software loaded therein, a method for allowing a video game player to create animated 3D images from captured 2D image data and interact with the animated 3D images to provide interactive game play, comprising: (a) providing a captured 2D color image to the home video game machine system for storage into said at least one memory; (b) allowing the video game player to select a portion of the captured 2D color image by operating the user-operable hand-held controller of the home video game machine system, the selected portion being at least temporarily stored in said at least one memory communicating with the 3D graphics coprocessor of the home video game machine system; (c) processing, with at least one of the home video game machine system main processor or the home video game machine system 3D graphics coprocessor, the selected 2D color image portion to convert the selected 2D color image portion into a color texture; (d) texture mapping, using the home video game machine system 3D graphics coprocessor, the color texture obtained from the processed selected 2D color image portion onto a predefined surface of a 3D video game character defined by plural polygon coordinates stored in the at least one memory communicating with the 3D graphics coprocessor; and (e) animating and displaying, using at least the main processor and 3D graphics coprocessor, the 3D video game character having the applied color texture to provide interactive video game play in response to manipulation of the user-operable hand-held controller.

☐ 15. Document ID: US 6614445 B1

L20: Entry 15 of 39 File: USPT Sep 2, 2003

DOCUMENT-IDENTIFIER: US 6614445 B1
TITLE: Antialiasing method for computer graphics

Brief Summary Text (6):

h e b b g e e e f e f ef b e

During the rasterization stage, the graphics processor renders each primitive into the frame buffer. The graphics processor accomplishes this task by determining which frame buffer memory locations are included within the bounds of each primitive. The included memory locations are then initialized to reflect the attributes of the primitive, including color and texture.

☐ 16. Document ID: US 6611272 B1

L20: Entry 16 of 39 File: USPT Aug 26, 2003

DOCUMENT-IDENTIFIER: US 6611272 B1
**** See image for Certificate of Correction ****
TITLE: Method and apparatus for rasterizing in a hierarchical tile order

Brief Summary Text (6):
During the rasterization stage, the graphics processor renders each primitive into the frame buffer. The graphics processor accomplishes this task by determining which frame buffer memory locations are included within the bounds of each primitive. The included memory locations are then initialized to reflect the attributes of the primitive, including color and texture.

Brief Summary Text (19):
The graphics processor rasterizes graphics primitives into the frame buffer. To accomplish this task, the graphics processor determines which frame buffer memory locations are included within the bounds of each primitive. The included memory locations are then initialized to reflect the attributes of the primitive, including color and texture. During rasterization, the graphics processor uses a hierarchy of memory tiles. Within this hierarchy, smaller tiles are grouped into larger tiles. These larger tiles may be grouped, in turn, into still larger tiles. For a representative embodiment of the present invention, the tile hierarchy includes three levels. The lowest level of the hierarchy is made up of four pixel by four pixel low-level tiles. These four-by-four tiles are grouped into eight-by-eight mid-level tiles and the eight-by-eight tiles are grouped into sixteen-by-sixteen high-level tiles.

☐ 17. Document ID: US 6587111 B2

L20: Entry 17 of 39 File: USPT Jul 1, 2003

DOCUMENT-IDENTIFIER: US 6587111 B2
TITLE: Graphic processor and data processing system

Detailed Description Text (11):
The rendering buffer unit 101 is used as a buffer between the unified memory unit 4 external to the graphic processor 1 and the rendering unit 100. To be more

h e b b g e e e f e f ef b e

specific, data read out from the memory-buffer area of the unified memory unit 4 and data to be written into the memory-buffer area is temporarily stored in the rendering buffer unit 101. In addition, information, such as a command and a texture, read out from the unified memory unit 4 are also temporarily stored in the rendering buffer unit 101. On the other hand, the display buffer unit 103 is used as a buffer between the unified memory unit 4 and the display unit 102. To be more specific, image information read out from the frame-buffer area of the unified memory unit 4 is temporarily stored in the display buffer unit 103 before being passed on to the display unit 102. The display buffer unit 103 has a color pallet 1031 for converting image data of 1 pixel per 8 bits into image data of 1 pixel per 16 bits.

☐ 18. Document ID: US 6466223 B1

L20: Entry 18 of 39 File: USPT Oct 15, 2002

DOCUMENT-IDENTIFIER: US 6466223 B1
**** See image for Certificate of Correction ****
TITLE: Method and apparatus for texture memory management

Brief Summary Text (6):
During the rasterization stage, the graphics processor renders each primitive into the frame buffer. The graphics processor accomplishes this task by determining which frame buffer memory locations are included within the bounds of each primitive. The included memory locations are then initialized to reflect the attributes of the primitive, including color and texture.

☐ 19. Document ID: US 6443842 B1

L20: Entry 19 of 39 File: USPT Sep 3, 2002

DOCUMENT-IDENTIFIER: US 6443842 B1
**** See image for Certificate of Correction ****
TITLE: Method, program product, and game system for blurring an image

Detailed Description Text (8):
The VRAM 17 of the graphics processor 16, as shown in FIG. 3, has defined in it two (#1 and #2) frame buffers 17A and 17B for storing pixel data for display on a screen, a Z-value buffer 17C for storing a Z-value showing a position in a depth direction in the virtual three-dimensional space for each pixel in the frame buffers 17A and 17B, an accumulation buffer 17D used for the blurring processing, and a texture buffer 17E for storing texture data required for polygon display. Note that the Z-value is larger closer up and smaller further away in the virtual three-dimensional space.

h e b b g e e e f e f ef b e

☐ 20. Document ID: US 6412061 B1

L20: Entry 20 of 39 File: USPT Jun 25, 2002

DOCUMENT-IDENTIFIER: US 6412061 B1
TITLE: Dynamic pipelines with reusable logic elements controlled by a set of multiplexers for pipeline stage selection

Brief Summary Text (9):
A graphics processor generally performs data transfer operations and functions for drawing points, lines, polylines, text, string text, triangles, and polygons to the frame buffer. Furthermore, the graphics processor performs many graphics functions on the data within the frame buffer, such as patterning, depth cueing, color compare, alpha blending, accumulation, texture assist, anti-aliasing, supersampling, color masking, stenciling, panning and zooming, error correction, as well as depth and color interpolation, among other functions.

Detailed Description Text (3):
The RAMDAC 108 receives digital data stored in a frame buffer 110 and converts the digital data to the appropriate analog outputs required by a display unit 112. In the preferred embodiment, the frame buffer 110 is part of a raster display implemented in a Video RAM (VRAM) organization by Texas Instruments, where the digital data comprises a bitmap representing a rectangular array of picture elements referred to as pixels or pixel values. Each pixel value defines the color of the corresponding pixel on a screen of the display unit 112, and each pixel value is preferably 24 bits for a full color display. The display unit 112 may be any type, such as a cathode ray tube (CRT) or a liquid crystal display (LCD) commonly used for portable computers. The transceivers 106 are used to interface the processor 100 with the system bus 102 through address and data signals, collectively referred to as the HBUS 114, which is further connected to an optional private memory 116. In the preferred embodiment, the private memory 116 acts as a virtual frame buffer, display list storage, texture map, and bitmapped font storage memory to improve performance and functionality of the graphics system. The private memory 116 is preferably added as a separate bank of external dynamic RAMs (DRAMs) for providing a performance improvement by permitting faster access to display list instructions and pixel data compared to data stored in main memory 126 of the host computer system. The graphics processor 100 communicates to the frame buffer 110 through address, data, and control lines, collectively referred to as the LBUS 118, which is further connected to a Z buffer 122, also preferably implemented using DRAMs. The Z buffer 122 is optional in a graphics system, and is preferably used to implement a depth buffer for three-dimensional (3D) graphic displays. Separate control signals 124 are also connected between the processor 100 and the Z buffer 122.

Full | Title | Citation | Front | Review | Classification | Date | Reference | Claims | KWIC | Draw De

☐ 21. Document ID: US 6409598 B1

L20: Entry 21 of 39 File: USPT Jun 25, 2002

h e b b g e e e f e f ef b e

DOCUMENT-IDENTIFIER: US 6409598 B1
** See image for Certificate of Correction **
TITLE: Method, program product, and game system for blurring an image

Detailed Description Text (8):
The VRAM 17 of the graphics processor 16, as shown in FIG. 2, has defined in it two (#1 and #2) frame buffers 17A and 17B for storing pixel data for display on a screen, a Z-value buffer 17C for storing a Z-value showing a position in a depth direction in the virtual three-dimensional space for each pixel in the frame buffers 17A and 17B, and a texture buffer 17D for storing texture data required for polygon display. Note that the Z-value is larger closer up and smaller further away in the virtual three-dimensional space.

Detailed Description Text (15):
FIG. 5 shows the flow of the depth-of-field processing. The depth-of-field processing is blurring processing. First, the graphics processor 16 reduces the images a, b, and c of the frame buffer 17A or 17B as shown schematically in FIG. 6A and FIG. 6B and copies the content to the texture buffer 17D (step S30). In this case, as shown in FIG. 7, the image a is at the frontmost side of the virtual three-dimensional space and the image b and image c are deeper in the three-dimensional space, in that order. Note that the minimum value Zmin to maximum value Zmax are blurring areas and may be set as parameters.

22. Document ID: US 6384831 B1

L20: Entry 22 of 39 File: USPT May 7, 2002

DOCUMENT-IDENTIFIER: US 6384831 B1
TITLE: Graphic processor and data processing system

Detailed Description Text (11):
The rendering buffer unit 101 is used as a buffer between the unified memory unit 4 external to the graphic processor 1 and the rendering unit 100. To be more specific, data read out from the memory-buffer area of the unified memory unit 4 and data to be written into the memory-buffer area is temporarily stored in the rendering buffer unit 101. In addition, information, such as a command and a texture, read out from the unified memory unit 4 are also temporarily stored in the rendering buffer unit 101. On the other hand, the display buffer unit 103 is used as a buffer between the unified memory unit 4 and the display unit 102. To be more specific, image information read out from the frame-buffer area of the unified memory unit 4 is temporarily stored in the display buffer unit 103 before being passed on to the display unit 102. The display buffer unit 103 has a color pallet 1031 for converting image data of 1 pixel per 8 bits into image data of 1 pixel per 16 bits. The CPU interface unit 105 comprises components including a DMA (Direct Memory Access) control circuit 1051 and a data-format converting circuit 1052. The data-format converting circuit 1052 converts a digital video format YUV or a digital video format .DELTA.YUV for a navigation system into an RGB (Red, Green and Blue) format. It should be noted that the symbol Y represents luminance while notation U/V is a chrominance difference component.

☐ 23. Document ID: US 6359626 B1

L20: Entry 23 of 39 File: USPT Mar 19, 2002

DOCUMENT-IDENTIFIER: US 6359626 B1
TITLE: Multisample dither method with exact reconstruction

Brief Summary Text (6):
During the rasterization stage, the graphics processor renders each primitive into the frame buffer. The graphics processor accomplishes this task by determining which pixels (i.e., which frame buffer memory locations) are included within the bounds of each primitive. The frame buffer memory locations for included pixels are then initialized to reflect the attributes of the primitive, including color and texture.

☐ 24. Document ID: US 6222550 B1

L20: Entry 24 of 39 File: USPT Apr 24, 2001

DOCUMENT-IDENTIFIER: US 6222550 B1
TITLE: Multiple triangle pixel-pipelines with span-range pixel interlock for processing separate non-overlapping triangles for superscalar 3D graphics engine

Abstract Text (1):
A 3D graphics processor has parallel triangle pixel pipelines. One or more triangle setup engine(s) receives triangle primitives from a host or geometry engine and generates vertex color, texture and other attributes as well as their gradients. The triangle setup engine makes available all required triangle data to the triangle pixel pipelines. The triangle pixel pipelines accept the next triangle data on a demand basis, when finished with the previous triangle. Each triangle pixel pipeline has a span engine that generates endpoints along the 3 edges of the triangle where the horizontal lines (spans) intersect. Each triangle pixel pipeline also has a raster engine that receives the endpoints as well as gradients and generates color, texture and other attributes for each pixel along a span between endpoints. The raster engine then composites pixels from these attributes and updates visible pixels in the frame buffer. Pixel-memory coherency for Z-buffering is maintained by comparing an MSB part of the X pixel address and the span line number (Y address) of pixels being processed in each pipeline. Thus a span-range of pixels is compared rather than individual pixels. When span-ranges of pixels being rasterized in two different triangle pixel-pipelines overlap, one of the pipelines must stall until the other finishes the span-range. For rendering modes with alpha-blending enabled or Z-buffering disabled triangle input order must be maintained during pixel rasterization. To maintain proper order, bounding boxes of triangles processed in different pipelines are compared. When the bounding boxes overlap, even if the triangles do not overlap, one pipeline is held until the other completes the triangle.

h e b b g e e e f e f ef b e

Full | Title | Citation | Front | Review | Classification | Date | Reference | Claims | KWIC | Draw De

☐ 25. Document ID: US 6219062 B1

L20: Entry 25 of 39 File: USPT Apr 17, 2001

DOCUMENT-IDENTIFIER: US 6219062 B1
TITLE: Three-dimensional graphic display device

Detailed Description Text (15):
FIG. 2 shows an example of the internal structure of the graphics processor 50. In operation, the graphics processor 50 gains access to a command list 300 in the main memory 30 to read figure drawing commands therefrom. According to the commands, the graphics processor 50 draws figures in the frame buffer area. Each figure is assigned color data on its vertexes, depth data and texture pattern data. It is on the basis of such data that the graphics processor 50 calculates the coordinates, color and depth of each pixel for drawing each figure.

Full | Title | Citation | Front | Review | Classification | Date | Reference | Claims | KWIC | Draw De

☐ 26. Document ID: US 6166724 A

L20: Entry 26 of 39 File: USPT Dec 26, 2000

DOCUMENT-IDENTIFIER: US 6166724 A
TITLE: Method and apparatus for sequencing palette updates in a video graphics system

Detailed Description Text (7):
FIG. 2 illustrates a video graphics processing system that includes processor 10, main memory 20, graphics processor 30, frame buffer 40, and bus 50 which couples the components of the system. The processor 10 may have addition connections to the main memory 20. Preferably, the graphics processor 30 and frame buffer 40 are dedicated to video graphics processing aspects of the system. The graphics processor 30 includes the lookup table 36, which can be used to store compressed color sets or textures.

Full | Title | Citation | Front | Review | Classification | Date | Reference | Claims | KWIC | Draw De

☐ 27. Document ID: US 6005580 A

L20: Entry 27 of 39 File: USPT Dec 21, 1999

DOCUMENT-IDENTIFIER: US 6005580 A

h e b b g e e e f e f ef b e

**** See image for Certificate of Correction ****
TITLE: Method and apparatus for performing post-process antialiasing of polygon edges

Detailed Description Text (56):
In response to receiving the state data from the processor 105 indicating the texture-mapped output image should be drawn, the graphics processor 710 (executing graphics card routines accessed from the memory 750) processes the geometry and texture map data and transmits the necessary pixel information to a pixel engine 730 to draw the texture mapped output image. In one embodiment, this pixel information includes the pixel coordinates (Px,Py,Pz), the pixel color (RGB), the alpha (A), the fog (F), the texture coordinates (u,v), and the perspective factor (Q). The pixel engine 730 includes a filter unit 735 and a draw unit 745. The filter unit 735 is coupled to the D-cache 720 to allow for access to the texture mapped images stored in the texture map memory area 775. Based on the texture coordinates received, the filter unit 735 modifies texels in the texture maps to perform antialiasing of those texture maps and transmits the results to the draw unit 745. The draw unit 745 uses the pixel coordinates and color data to draw the texture mapped output image through the D-cache 720 to the memory 750. In one embodiment, the frame buffer memory area 770 can be both read and written. In the embodiment in which the frame buffer memory area 770 can be both read and written, the texture mapped output image is written to the frame buffer memory area 770. Alternative embodiments may store the texture mapped output image in any number of different storage areas. In addition, the storage area used to store the texture mapped output image may be located any number of places, including in the graphics card, in the monitor, etc. When drawing polygons, the draw unit 745 may be implemented to perform any number of functions, such as blending, hazing, and fogging.

☐ 28. Document ID: US 5969728 A

L20: Entry 28 of 39 File: USPT Oct 19, 1999

DOCUMENT-IDENTIFIER: US 5969728 A
TITLE: System and method of synchronizing multiple buffers for display

Detailed Description Text (4):
The memory 118 is a separate bank of external random access memory (RAM) devices or the like, such as dynamic RAM (DRAM), synchronous RAM (SRAM), EDO RAM, RDRAM, etc., coupled to the graphics processor 114 via one or more memory channels 124. In the preferred embodiment, the memory 118 includes up to 32 Megabytes (MB) of RDRAM for storing a frame buffer 119 used for drawing and display purposes. The frame buffer 119 includes a description of each pixel on the display unit 122. A rectangular portion of the frame buffer 119 is referred to as the display rectangle, which is "visible" on the display unit 122. The format of the pixel and texel descriptions in the frame buffer 119 is either in palletized, Red, Green and Blue (RGB) or YUV format or any combination of these formats. The memory 118 may also include an off-screen color buffer, Z buffer and texture maps. The frame buffer 119 is further described below.

☐ 29. Document ID: US 5793386 A

L20: Entry 29 of 39 File: USPT Aug 11, 1998

DOCUMENT-IDENTIFIER: US 5793386 A
TITLE: Register set reordering for a graphics processor based upon the type of primitive to be rendered

Detailed Description Text (2):
Referring now to FIG. 1, the present invention relates generally to a graphics system for a personal computer (PC) capable of rendering points, lines and polygons using a main slope technique. As shown in FIG. 1, the graphics system generally includes a host processor 50 and system memory 75 coupled to a system bus 25, a graphics processor 100, a frame buffer, such as an RDRAM array 85, and a display unit 60. The host processor 50 may comprise the central processing unit of a PC, while system memory 75 may comprise the working memory, or random access memory array of the PC. The host processor 50 preferably includes a software driver for generating display parameters, which describe display objects to be rendered by the graphics processor 100. Thus, for example, the software driver may identify the spatial location of points, line endpoints, or vertex coordinates of polygons. For polygons, the software driver also preferably identifies main slope and width slope values for the polygon. When applicable, the software driver may identify color intensity, texture, and slope values, and various other parameter values as will be understood by one skilled in the art. Thus, the software driver calculates and loads main and orthogonal slopes, start and stop values for pixel position, intensity, depth and transparency of objects to be rendered by the graphics processor 100. The host processor or software driver also generates an operational code instruction which defines the type of primitive to be rendered. The software driver preferably is loaded into the system memory 75 from a permanent magnetic storage device, such as a hard drive or CD ROM drive device. Once loaded, the software driver is executed by the host processor 50.

☐ 30. Document ID: US 5790134 A

L20: Entry 30 of 39 File: USPT Aug 4, 1998

DOCUMENT-IDENTIFIER: US 5790134 A
TITLE: Hardware architecture for image generation and manipulation

Brief Summary Text (14):
The degree of pixel processing performed by the rendering processor (and not other hardware or software components) can vary with desired system capability. At a minimum, the rendering processor is capable of drawing pixels into the frame buffer in response to commands received from the host CPU, a geometry processor or other high-level graphics processor; this entails computing actual pixel addresses in the frame buffer from X and Y values or geometric specifications provided by the CPU.

h e b b g e e e f e f ef b e

The rendering processor preferably also performs interpolation operations to determine individual pixel values from end-point coordinates or their equivalent. In addition, the rendering processor can be provided with the ability to perform special processing algorithms and mathematical operations, such as antialiasing and dithering; alpha blending; Z-buffering; fog computations (which add white to a pixel image value to simulate fog and thereby provide depth cueing); clipping to a window or other boundary; double buffering of the image (i.e., generating an image while keeping in memory the currently displayed, previously generated image); and texture-map processing. Once again, the functionality with which the rendering processor is provided depends on the relative benefit (in terms of economy and time performance) of withdrawing particular functions from software or other hardware; ideally, the choice will maximize throughput over the main system bus and minimize latencies.

31. Document ID: US 5778250 A

L20: Entry 31 of 39 File: USPT Jul 7, 1998

DOCUMENT-IDENTIFIER: US 5778250 A
TITLE: Method and apparatus for dynamically adjusting the number of stages of a multiple stage pipeline

Brief Summary Text (9):
A graphics processor generally performs data transfer operations and functions for drawing points, lines, polylines, text, string text, triangles, and polygons to the frame buffer. Furthermore, the graphics processor performs many graphics functions on the data within the frame buffer, such as patterning, depth cueing, color compare, alpha blending, accumulation, texture assist, anti-aliasing, supersampling, color masking, stenciling, panning and zooming, error correction, as well as depth and color interpolation, among other functions.

Detailed Description Text (3):
The RAMDAC 108 receives digital data stored in a frame buffer 110 and converts the digital data to the appropriate analog outputs required by a display unit 112. In the preferred embodiment, the frame buffer 110 is part of a raster display implemented in a Video RAM (VRAM) organization by Texas Instruments, where the digital data comprises a bitmap representing a rectangular array of picture elements referred to as pixels or pixel values. Each pixel value defines the color of the corresponding pixel on a screen of the display unit 112, and each pixel value is preferably 24 bits for a full color display. The display unit 112 may be any type, such as a cathode ray tube (CRT) or a liquid crystal display (LCD) commonly used for portable computers. The transceivers 106 are used to interface the processor 100 with the system bus 102 through address and data signals, collectively referred to as the HBUS 114, which is further connected to an optional private memory 116. In the preferred embodiment, the private memory 116 acts as a virtual frame buffer, display list storage, texture map, and bitmapped font storage memory to improve performance and functionality of the graphics system. The private memory 116 is preferably added as a separate bank of external dynamic RAMs (DRAMs) for providing a performance improvement by permitting faster access to display list instructions and pixel data compared to data stored in main memory 126 of the host computer system. The graphics processor 100 communicates to the frame buffer 110 through address, data, and control lines, collectively referred to as the LBUS 118, which is further connected to a Z buffer 122, also preferably implemented using

h e b b g e e e f e f ef b e

DRAMs. The Z buffer 122 is optional in a graphics system, and is preferably used to implement a depth buffer for three-dimensional (3D) graphic displays. Separate control signals 124 are also connected between the processor 100 and the Z buffer 122.

32. Document ID: US 5740344 A

L20: Entry 32 of 39 File: USPT Apr 14, 1998

DOCUMENT-IDENTIFIER: US 5740344 A
TITLE: Texture filter apparatus for computer graphics system

Detailed Description Text (2):
FIG. 5 depicts a computer system 100 according to an embodiment of the present invention. As before, the computer system includes a processor or CPU 112, a main memory 114, a disk memory 116, an input device 118, a bus 120, and a graphics processor 130. The graphics processor 130 includes a drawing processor 132, a frame buffer 134, an address generator 136 and a texture color interpolator 200. A display device 138 is also connected to the frame buffer 134. As before, the drawing processor 132 receives instructions, and information (stored in the main memory 114 or disk memory 116) regarding objects in 3-D space. The drawing processor 132 renders such objects in 3-D space, i.e., draws pixel representations of the images of the objects and then maps texture onto the surface of each object. In mapping texture onto object surfaces, the drawing processor 132 utilizes the texture color interpolator 200 to interpolate texture color values. The images thus produced are stored in frames of the frame buffer 134. The frames in the frame buffer are then displayed on the display device 138.

33. Document ID: US 5684941 A

L20: Entry 33 of 39 File: USPT Nov 4, 1997

DOCUMENT-IDENTIFIER: US 5684941 A
TITLE: Interpolation rendering of polygons into a pixel grid

Detailed Description Text (4):
The transceivers 106 are used to interface the graphics processor 100 with the system bus 102 through address and data signals, collectively referred to as the HBUS 114, which is further connected to an optional private memory 116. In the preferred embodiment, the private memory 116 acts as a virtual frame buffer, display list storage, texture map, and bit mapped fonts storage memory to improve performance and functionality of the graphics system. The private memory 116 is preferably added as a separate bank of external dynamic RAMs (DRAMs) for providing a performance improvement by permitting faster access to display list instructions and pixel data compared to data stored in main memory 126 of the host computer

h e b b g e e e f e f ef b e

system. The graphics processor 100 communicates to the frame buffer 110 through address data and control lines, collectively referred to as the LBUS 118, which is further connected to a Z-buffer 122, also preferably implemented using DRAMs. The Z-buffer 122 is preferably used to implement a depth buffer for three-dimensional (3D) graphic displays. Separate control signals 124 are also connected between the graphics processor 100 and the Z-buffer 122.

☐ 34. Document ID: US 5664162 A

L20: Entry 34 of 39 File: USPT Sep 2, 1997

DOCUMENT-IDENTIFIER: US 5664162 A
TITLE: Graphics accelerator with dual memory controllers

Abstract Text (1):
A processor having two separate and relatively independent memory controllers to achieve a dual interface architecture. A first memory controller is coupled to the host interface for retrieving data and instructions and a second memory controller is coupled to an independent local bus for interfacing with a frame buffer memory. A depth buffer may also be coupled to the local bus if desired. Address multiplexor logic is preferably included to allow either memory controller to address either external bus. Multiplexor and buffer logic is also preferably included to allow data transfer in either direction. Preferably, the processor is a graphics processor and both memory controllers are programmable for different addressing formats, such as linear and X/Y in the preferred embodiment. In this manner, data is transferred from host to local memories, and vice versa, in any desired format without delays due to memory controller reconfiguration. Data transfers from one location to another within a single memory, such as window moves within the frame buffer, are achieved much faster. Dual memory controllers allow command or instruction prefetching during execution of a previous command. More sophisticated graphics functions, such as texture mapping and data alignment, are also performed much faster and more efficiently.

☐ 35. Document ID: US 5649173 A

L20: Entry 35 of 39 File: USPT Jul 15, 1997

DOCUMENT-IDENTIFIER: US 5649173 A
TITLE: Hardware architecture for image generation and manipulation

Brief Summary Text (14):
The degree of pixel processing performed by the rendering processor (and not other hardware or software components) can vary with desired system capability. At a minimum, the rendering processor is capable of drawing pixels into the frame buffer in response to commands received from the host CPU, a geometry processor or other

h e b b g e e e f e f ef b e

high-level graphics processor; this entails computing actual pixel addresses in the frame buffer from X and Y values or geometric specifications provided by the CPU. The rendering processor preferably also performs interpolation operations to determine individual pixel values from end-point coordinates or their equivalent. In addition, the rendering processor can be provided with the ability to perform special processing algorithms and mathematical operations, such as antialiasing and dithering; alpha blending; Z-buffering; fog computations (which add white to a pixel image value to simulate fog and thereby provide depth cueing); clipping to a window or other boundary; double buffering of the image (i.e., generating an image while keeping in memory the currently displayed, previously generated image); and texture-map processing. Once again, the functionality with which the rendering processor is provided depends on the relative benefit (in terms of economy and time performance) of withdrawing particular functions from software or other hardware; ideally, the choice will maximize throughput over the main system bus and minimize latencies.

☐ 36. Document ID: US 5625768 A

L20: Entry 36 of 39 File: USPT Apr 29, 1997

DOCUMENT-IDENTIFIER: US 5625768 A
TITLE: Method and apparatus for correcting errors in pixel characteristics when interpolating polygons into a pixel grid

Detailed Description Text (4):
The transceivers 106 are used to interface the graphics processor 100 with the system bus 102 through address and data signals, collectively referred to as the HBUS 114, which is further connected to an optional private memory 116. In the preferred embodiment, the private memory 116 acts as a virtual frame buffer, display list storage, texture map, and bit mapped fonts storage memory to improve performance and functionality of the graphics system. The private memory 116 is preferably added as a separate bank of external dynamic RAMs (DRAMs) for providing a performance improvement by permitting faster access to display list instructions and pixel data compared to data stored in main memory 126 of the host computer system. The graphics processor 100 communicates to the frame buffer 110 through address data and control lines, collectively referred to as the LBUS 118, which is further connected to a Z-buffer 122, also preferably implemented using DRAMs. The Z-buffer 122 is preferably used to implement a depth buffer for three-dimensional (3D) graphic displays. Separate control signals 124 are also connected between the graphics processor 100 and the Z-buffer 122.

☐ 37. Document ID: US 5469535 A

L20: Entry 37 of 39 File: USPT Nov 21, 1995

DOCUMENT-IDENTIFIER: US 5469535 A

h e b b g e e e f e f ef b e

TITLE: Three-dimensional, texture mapping display system

Detailed Description Text (2):
Referring now in detail to the drawings, there is shown in FIG. 1 an overall block diagram of an improved three-dimensional, texture mapping display system 10 constructed in accordance with the principles of the present invention. The display system is used to display three-dimensional projected polygons with texture maps on a two-dimensional raster display device. The display system 10 is particularly adapted for use in video games to allow real-time animation of video game scenes with textured plane surfaces at a high speed of operation. The display system is comprised of a host computer 12 which is preferably a digital signal processor (DSP), a graphics co-processor 14, a texture memory 16, a video frame buffer 18, a color look-up table 20, and a cathode ray tube (CRT) or other display device 22.

Full | Title | Citation | Front | Review | Classification | Date | Reference | | | Claims | KWIC | Draw De

☐ 38. Document ID: JP 2002063594 A

L20: Entry 38 of 39 File: JPAB Feb 28, 2002

DOCUMENT-IDENTIFIER: JP 2002063594 A
TITLE: GRAPHICS SYSTEM WITH COPY OUT CONVERSION BETWEEN EMBEDDED FRAME BUFFER AND MAIN MEMORY

Abstract Text (2):
SOLUTION: A graphics processor has an embedded frame buffer, and a copy line is provided between the embedded frame buffer 114 and an external frame buffer such as a main memory 112 connected directly to a display device 56. The copy line performs four types of combination conversion of data from a certain format into another format, for instance, one RGB and YUV color format into another RGB and YUV color format. The converted data are transferred to the external frame buffer or a texture buffer for being used as texture.

Full | Title | Citation | Front | Review | Classification | Date | Reference | | | Claims | KWIC | Draw De

☐ 39. Document ID: EP 1182617 A2

L20: Entry 39 of 39 File: EPAB Feb 27, 2002

PUB-NO: EP001182617A2
DOCUMENT-IDENTIFIER: EP 1182617 A2
TITLE: Graphics system with reconfigurable embedded frame buffer and copy out conversions between embedded frame buffer and main memory

Full | Title | Citation | Front | Review | Classification | Date | Reference | | | Claims | KWIC | Draw De

Clear | Generate Collection | Print | Fwd Refs | Bkwd Refs | Generate OACS

h e b b g e e e f e f ef b e

Terms	Documents
graphics $5processor same frame buffer same (texture) same (color or depth)	39

Display Format: - Change Format

Previous Page Next Page Go to Doc#